L5: Entry 6 of 12 File: TDBD Aug 1, 1988

TDB-ACC-NO: NN8808303

DISCLOSURE TITLE: High Performance Microprocessor Memory System

PUBLICATION-DATA:
IBM Technical Disclosure Bulletin, August 1988, US

VOLUME NUMBER: 31
ISSUE NUMBER: 3
PAGE NUMBER: 303 - 306

PUBLICATION-DATE: August 1, 1988 (19880801)

CROSS REFERENCE: 0018-8689-31-3-303

DISCLOSURE TEXT:

- This article describes a memory system that improves the performance of the memory access time to match processor performance. Microprocessor performance has been increasing at a greater rate than that of associated memory systems. The memory designs to solve this problem have usually involved expensive, high-performance memory components or elaborate, multilevel cache design. ***** SEE ORIGINAL DOCUMENT ***** The system being described combines fast bipolar logic circuits with slower, inexpensive memory components so as to provide fast memory access times at a low cost. The design requires only a single level of memory. It uses simple prefetch and queueing logic located in the memory system instead of in the controller or processor so that processor and memory cycles can be overlapped to a greater extent. Fig. 1 is a block diagram showing a single gate-array memory controller incorporating two prefetch queues for memory read operations and a single queue for memory write operations. Fig. 1 includes two byte-wide data paths (BYTE 0 and BYTE 1). In each data path is a prefetch queue data register (QUE1 and QUE2), a buffer register (used as temporary storage for data to be written or read), and a four-way multiplexer used to load each buffer register from the prefetch queues, from the memory bus, or from the system bus. Also shown are ECC (Error Correcting Code) checking and generating components and spare memory module swapping multiplexers. Fig. 2 shows the memory controller 20-bit address path. Included in the address path are a 20-bit incrementer, a temporary address register (to hold the next address to be accessed during either a queue loading operation or a write operation), two 20-bit prefetch address registers (QUE1 ADDR and QUE2 ADDR), and two 20-bit comparator components (COMP1 and COMP2). Also shown are a refresh counter and multiplexers required for dynamic memory address control. The prefetch queues are implemented as single level read queues. Each queue includes a data register and a register containing the address from which the data was originally read. Whenever a memory read access is made, the memory controller determines whether the data requested is available in a prefetch queue data register by comparing the addresses in the prefetch queue address registers to the present address. If an address match is found, the desired data is in the queue data register corresponding the matching address register. If the data is available, the address is incremented and immediately placed on the memory address bus to refill the read queue. If the requested data is not available in a prefetch queue data register, an internal memory read cycle is initiated to access the data. The memory controller will cause this initial memory read cycle to be followed by a second sequential memory read to fill the prefetch queue. The prefetch queue to be filled with the new read data depends on the memory application. Several options are available. When the memory is used as an instruction control storage, one queue is associated with instruction reads and all instruction accesses are made via this specified queue. Data reads are made using the second read queue. When the memory is used as a shared data memory,

both read queues are used for data reads. When neither read queue contains the requested data, the least recently used queue is selected to store the new read data. Either or both of the read queues can be disabled if the specific application is known to require completely random read accesses of the memory. The memory write queue is also a single stage register queue. Whenever a memory write access is made, the memory controller latches the data and address to be written in the write queue registers and initiates an internal memory write cycle to place the data in memory. The memory write cycle is independent from the memory bus cycle being performed. Thus, the bus cycle can terminate quickly while the longer memory write cycle continues. The memory read and write queue controllers are also independent from one another to allow sequential memory reads from one or two different ranges of addresses while writing to a third range of addresses. When a microprocessor instruction stream repetitively cycles through the same range of addresses and the range is small enough to fit within the high speed cache memory, a cache memory system provides faster access time. Such faster cache access is available only on the second and subsequent accesses to the same address. Because the memory queuing scheme of this design prefetches data prior to any memory access, the data will be available in the high-speed buffer registers for any data reference including the first. Therefore, the faster memory access will be available for the initial accesses of most memory locations. The present design can be advantageously used in combination with a microprocessor cache system to provide performance enhancements beyond those offered by the cache system alone. The memory interface circuit includes logic which estimates the probable addresses of data that will be next accessed by the microprocessor. This data is fetched by the memory subsystem prior to the actual microprocessor memory access. The prefetched data is placed in one of several high speed data registers. When the microprocessor access actually occurs, the data may be read from a data register much faster than it was originally read from the memory devices. For microprocessor instruction execution streams, the prefetch data queues have been found to contain the required data approximately 90 percent of the time.

FIG. 1

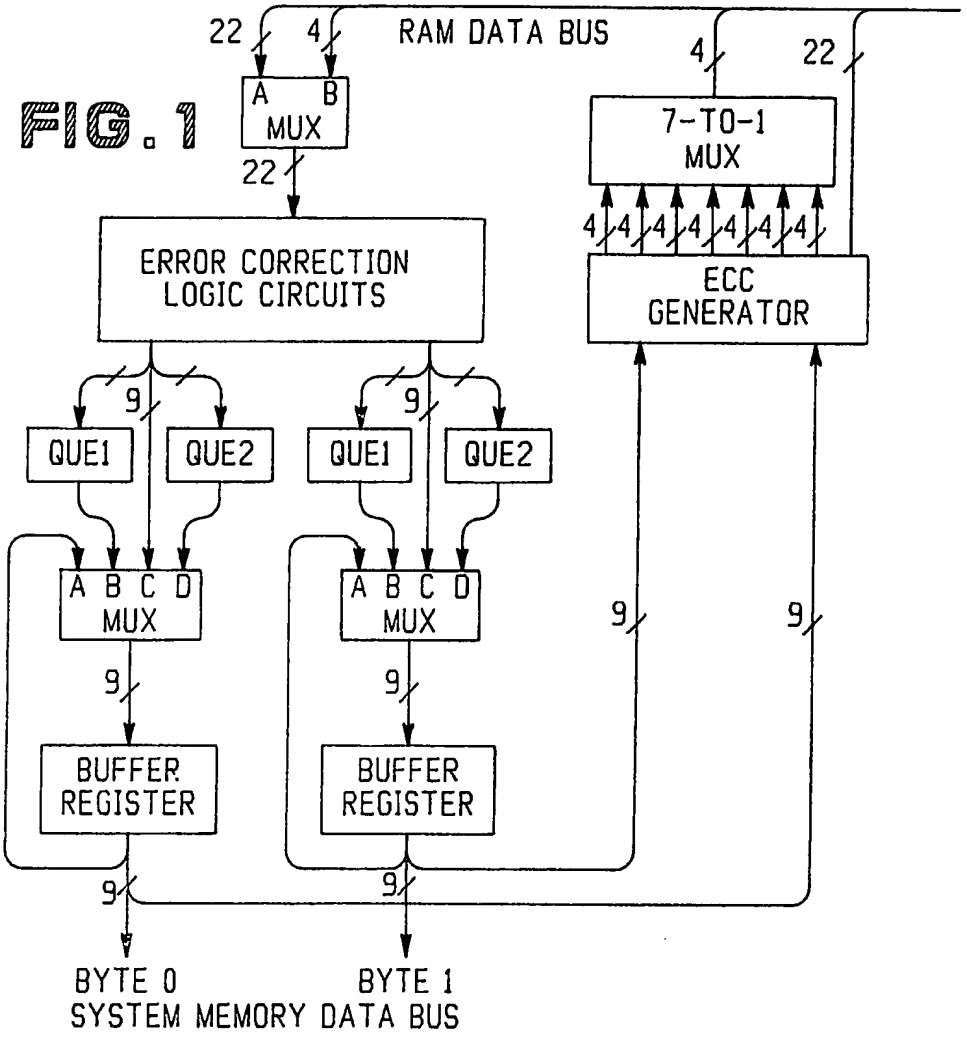
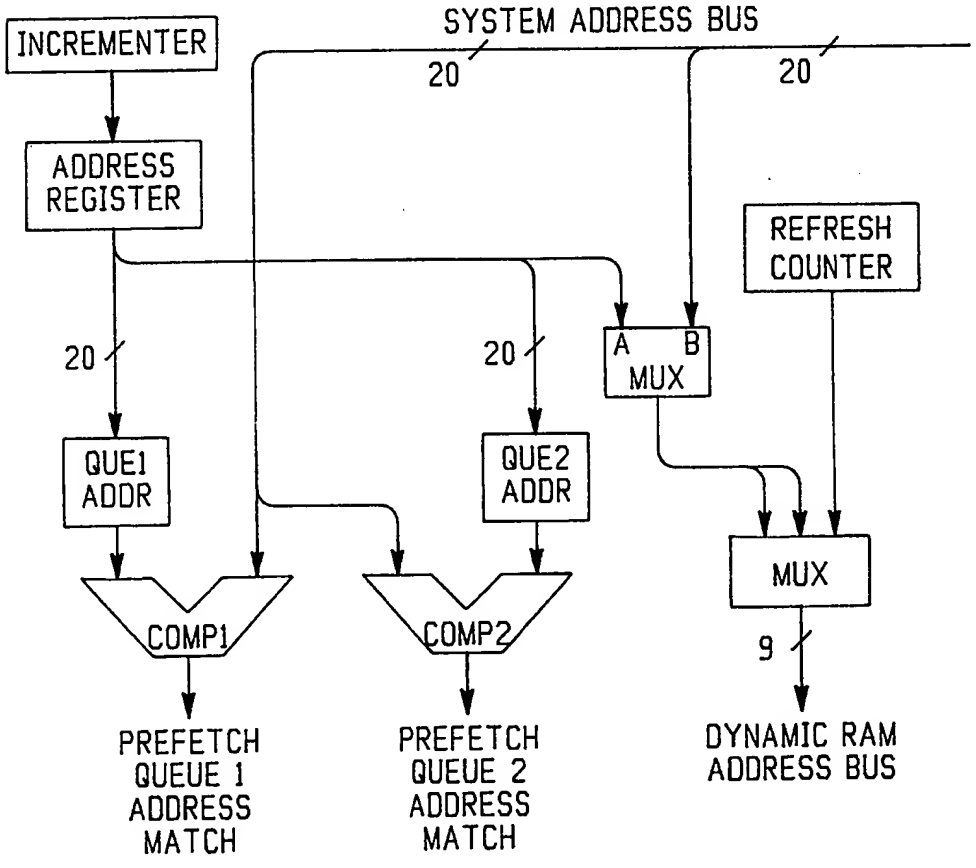

22
4
RAM DATA BUS
A B
MUX
22
4
22
7-TO-1
MUX
4 4 4 4 4 4 4
ERROR CORRECTION
LOGIC CIRCUITS
ECC
GENERATOR
9
9
QUE1
QUE2
QUE1
QUE2
A B C D
MUX
A B C D
MUX
9
9
9
9
BUFFER
REGISTER
BUFFER
REGISTER
9
9
BYTE 0
SYSTEM MEMORY DATA BUS
BYTE 1

FIG. 2


INCREMENTER
SYSTEM ADDRESS BUS
20
20
ADDRESS
REGISTER
REFRESH
COUNTER
20
20
A B
MUX
QUE1
ADDR
QUE2
ADDR
MUX
COMP1
COMP2
9
PREFETCH
QUEUE 1
ADDRESS
MATCH
PREFETCH
QUEUE 2
ADDRESS
MATCH
DYNAMIC RAM
ADDRESS BUS